Amendments to the Claims

Please amend claims 1, 23, 26, and 44; cancel claims 11, 15, 22, 33, 37-43, 54, 58-66; and add new claims 67-69, all as shown below. All pending claims are reproduced below, including those that remain unchanged.

1. (Currently Amended) A computer-implemented system to marshal and unmarshal data between XML and an object-oriented programming language, comprising:

~~an XML data;~~

an XML schema which defines [[the]] an XML data; and

a compiler, running on one or more processors of the computer-implemented system, to generate an object-oriented programming language type from the XML schema,

wherein the [[an]] object-oriented programming language type ~~which~~ corresponds to the XML schema and provides XML-oriented data manipulation, wherein the object-oriented programming language type allows the combination of a XML type system and an object-oriented programming language type ~~systems~~ system and ~~is capable of accessing and manipulating~~ can access and manipulate the XML data from within the object-oriented programming language,

wherein the object-oriented programming language type executes one or more XML data operations provided by the XML type system, on the XML data, to generate one or more result sets in the object-oriented programming language type system, wherein each of the one or more XML data operations is one of

an XML data query operation;

an XML data transformation operation; and

an XML data iteration operation.

~~; and~~

~~a compiler capable of generating the object-oriented programming language type from the XML schema, wherein the compiler runs on one or more processors.~~

2. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the compiler is capable of generating the object-oriented programming language type based on the definition of a web service method.

3. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the compiler is capable of generating the object-oriented programming language type based on a definition file.

4. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the compiler is capable of compiling an object-oriented programming language project into one or more regular ~~Java~~ object-oriented programming language types.

5. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type can be a movable cursor, capable of reading anywhere within the XML data.

6. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type can be [[a]] an immovable value, capable of referencing a fixed part of the XML data.

7. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type can be shared among multiple object-oriented programming language components.

8. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type is capable of updating the XML data within the object-oriented programming language.

9. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type is capable of accessing and updating object-oriented programming language data using object-oriented programming language type methods.

10. (Previously Presented) The computer-implemented system according to claim 1, wherein:

the object-oriented programming language type is capable of accessing and updating a database.

11. (Canceled).

12. (Previously Presented) The computer-implemented system according to claim 1, further comprising:

an XML schema capable of defining the legal types of the XML data, which include constraints on data types and ranges of the XML data; and constraints on the data types and ranges of the object-oriented programming language type.

13. (Previously Presented) The computer-implemented system according to claim 12, wherein:

the compiler is capable of generating constraints on the object-oriented programming language type from the XML schema on legal types of the XML data.

14. (Previously Presented) The computer-implemented system according to claim 12, wherein:

the constraints on the object-oriented programming language type are capable of validating the object-oriented programming language type.

15 – 22. (Canceled).

23. (Currently Amended) A method to marshal and unmarshal data between XML and an object-oriented programming language, comprising:

defining an XML data using an XML schema;

generating, via a compiler running on one or more processors, an object-oriented programming language type from the XML schema, wherein the ~~accessing from within object-oriented programming language elements of the XML data via an~~ object-oriented programming language type ~~which~~ corresponds to the XML schema and provides XML-oriented data manipulation, wherein the object-oriented programming language type extends from a base type that allows the combination of XML and object-oriented programming language type systems and can access and manipulate the XML data from within the object-oriented programming language; and

executing, via the object-oriented programming language type, one or more XML data operations provided by the XML type system, on the XML data, to generate one or more result sets in the object-oriented programming language type system, wherein each of the one or more XML data operations is one of

an XML data query operation;

an XML data transformation operation; and

an XML data iteration operation.

~~generating the object-oriented programming language type from the XML schema using a compiler.~~

24. (Previously Presented) The method according to claim 23, further comprising:

generating the object-oriented programming language type based on the definition of a web service method.

25. (Previously Presented) The method according to claim 23, further comprising:

generating the object-oriented programming language type based on a definition file.

26. (Currently Amended) The method according to claim 23, further comprising:

compiling ~~a Java~~ an object-oriented programming language project into one or more regular object-oriented programming language types.

27. (Previously Presented) The method according to claim 23, further comprising:

utilizing the object-oriented programming language type as a movable cursor to read anywhere within the XML data.

28. (Previously Presented) The method according to claim 23, further comprising:

utilizing the object-oriented programming language type as [[a]] an immovable value to reference a fixed part of the XML data

29. (Previously Presented) The method according to claim 23, further comprising:

sharing the object-oriented programming language type among multiple ~~Java~~ object-oriented programming language components.

30. (Previously Presented) The method according to claim 23, further comprising:

updating the XML data within an object-oriented programming language via the object-oriented programming language type.

31. (Previously Presented) The method according to claim 23, further comprising:

accessing and updating object-oriented programming language data using object-oriented programming language type methods.

32. (Previously Presented) The method according to claim 23, further comprising:

accessing and updating a database via the object-oriented programming language type.

33. (Canceled).

34. (Original) The method according to claim 23, further comprising:

defining the legal types of the XML data via an XML schema, which include constraints on data types and ranges of the XML data.

35. (Previously Presented) The method according to claim 34, further comprising:

generating constraints on the data types and ranges of the object-oriented programming language type from the XML schema on legal types of the XML data.

36. (Previously Presented) The method according to claim 34, further comprising:

validating the object-oriented programming language type using the constraints on the object-oriented programming language type.

37 – 43. (Canceled).

44. (Currently Amended) A machine readable storage medium having instructions stored thereon that when executed by a processor cause a system to:

define an XML data using an XML schema;

generate, via a compiler running on one or more processors, an object-oriented programming language type from the XML schema, wherein the ~~access the XML data via an object-oriented programming language type which corresponds to the XML schema and provides XML-oriented data manipulation, wherein the~~ object-oriented programming language type corresponds to the XML schema and provides XML-oriented data manipulation, wherein the object-oriented programming language type extends from a base type that allows the combination of XML and object-oriented programming language type systems and ~~is capable of accessing and manipulating the XML data from within an object-oriented programming language;~~ can access and manipulate the XML data from within the object-oriented programming language; and

execute, via the object-oriented programming language type, one or more XML data operations provided by the XML type system, on the XML data, to generate one or more result sets in the object-oriented programming language type system, wherein each of the one or more XML data operations is one of

an XML data query operation;

an XML data transformation operation; and

an XML data iteration operation.

~~generate the object-oriented programming language type from the XML schema using a compiler.~~

45. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

generate the object-oriented programming language type based on the definition of a web service method.

46. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

generate the object-oriented programming language type based on a definition file.

47. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

compile an object-oriented programming language project into one or more regular object-oriented programming language types with the compiler.

48. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

utilize the object-oriented programming language type as a movable cursor to read anywhere within the XML data.

49. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

utilize the object-oriented programming language type as an immovable value to reference a fixed part of the XML data.

50. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

share the object-oriented programming language type among multiple object-oriented programming language components.

51. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

update the XML data within an object-oriented programming language via the object-oriented programming language type.

52. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

access and update object-oriented programming language data using regular object-oriented programming language type methods.

53. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

access and update a database via the object-oriented programming language type.

54. (Canceled).

55. (Previously Presented) The machine readable storage medium of claim 44, further comprising instructions that when executed cause the system to:

define the legal types of the XML data via an XML schema, which include constraints on data types and ranges of the XML data.

56. (Previously Presented) The machine readable storage medium of claim 55, further comprising instructions that when executed cause the system to:

generate constraints on the object-oriented programming language type from the XML schema on legal types of the XML data.

57. (Currently Amended) The machine readable storage medium of claim 55, further comprising instructions that when executed cause
the system to:

validate the object-oriented programming language type using the constraints on the object-oriented programming language type.

58 - 66. (Canceled).

67. (New) The computer-implemented system according to claim 1, further comprising:

automatically generating the object-oriented programming language type for an object-oriented programming language component as an inner class to an XML control interface for the object-oriented programming language component.

68. (New) The computer-implemented system according to claim 1, further comprising:

sharing an object-oriented programming language type among multiple object-oriented programming language components, by explicitly referring the object-oriented programming language type when an object-oriented programming language component is defined.

69. (New) The system according to claim 1, wherein:

the object-oriented programming language type is both validated by one or more schema in XML type system and checked under the object-oriented programming language type system.